REMARKS

Claims 1-15, 17-20, 22-28, and 30-33 were pending. Claims 1, 10-14, 20, 23, 25, 26, 28, and 30-33 have been amended to clarify the nature of the invention. Claims 34-36 have been added. Support for the amendments to claims 1, 14, 20, 23, 26, 28, and 31 may be found in the Specification at least at paragraph [0080]. Support for the amendments to claims 10-13, 25 and new claims 34-36 may be found in the Specification at least at paragraph [0068] – [0074]. Accordingly, claims 1-15, 17-20, 22-28, and 30-36 remain pending subsequent entry of the present amendment.

35 U.S.C. § 101 Rejections

Claims 28-33 stand rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Applicant respectfully requests removal of these rejections in view of the present amendments to claims 28 and 30-33.

35 U.S.C. § 102 Rejection - Cohen

Claims 1-3, 10, 23, 28, and 31 are rejected under 35 U.S.C. 102(b) as being anticipated by Cohen (patent No. 5,115,506, hereinafter "Cohen"). Applicant respectfully traverses these rejections and requests reconsideration in view of the following remarks.

In the present Office Action, on page 3, paragraphs 6-7, it is suggested regarding claim 1 that:

> "Cohen taught the invention as claimed including . . . an interrupt vector generator for generating an exception vector associating with the interrupt handler, when the processing system receives and interrupt (e.g., see fig, 2 and col. 3, line 64-col. 4, line 8); mapping logic coupled to both of the register sets and the interrupt vector generator for selecting one of said plurality of register sets to be used by the interrupt handler (e.g., see col. 3,

> lines 13-45) wherein the mapping logic is programmably provided with a correlation between the exception vector and the selected one of the plurality of register sets (e.g., see col. 3, lines 13-45)."

However, Applicant submits that claim 1 recites features neither taught nor suggested by Cohen. Claim 1, reproduced below for reference purposes, recites:

> A processing system comprising:
> a plurality of shadow register sets;
> an interrupt vector generator, for generating an exception vector associating with an interrupt handler, when the processing system receives an interrupt;
> shadow set mapping logic, coupled to both said plurality of shadow register sets, and said interrupt vector generator, for selecting one of said plurality of shadow register sets to be used by said interrupt handler;
> wherein said shadow set mapping logic comprises a plurality of entries, each of which is programmable to associate exception vectors with at least one of said plurality of shadow register sets.

It is noted that the recited shadow set mapping logic includes features such as "a plurality of entries, each of which is programmable to associate exception vectors with at least one of said plurality of shadow register sets." Cohen discloses no such features. In contrast, Cohen merely discloses "hardware" determines which register set to use. More specifically, Cohen discloses:

> "The alternate register set 18 duplicates the registers in the normal register set 16. The CPU will access and use either the normal register set 16 or the alternate register set 18, but never both register sets at the same time. From the viewpoint of a user writing software for the CPU, which register set is being used is relatively transparent. That is, the user may simply direct that a particular instruction use data register D2 as a source of an operand or destination of an operation. Hardware will determine whether the D2 register of the normal set 16 or the D2 register of the alternate set 18 will be accessed." (Cohen, col. 3, lines 11-22).
>
> "Either the A/N bit 34 or the A/N' bit 36 indicates whether the CPU 14 is presently using the normal register set 16 or the alternate register set 18. A "zero" value indicates that the alternate register set 18 is being used and a

> "one" value indicates that the normal register set 16 is being used." (Cohen, col. 3, line 46 – col. 4, line 1).

As may be seen from the above, Cohen merely discloses that "hardware" determines whether the normal or the alternate register set will be used and an A/N bit that indicates which register is presently being used. However, Cohen does not disclose mapping logic as recited, including plural entries for performing the determination, nor does Cohen suggest anything that is programmable to associate exception vectors with register sets. Accordingly, Applicant finds no teaching or suggestion in Cohen that "shadow set mapping logic comprises a plurality of entries, each of which is programmable to associate exception vectors with at least one of said plurality of shadow register sets," as is recited in claim 1. For at least these reasons, Applicant submits claim 1 is patentably distinguished from Cohen. As independent claims 23 and 31 include features similar to that of claim 1, claims 23 and 31 are patentably distinguished as well. Likewise, as each of dependent claims 2-13, 24, 25, 32, and 33-36 includes the features of the independent claims upon which it depends, each of claims 2-13, 24, 25, 32, and 33-36 is believed patentable for at least the reasons above as well.

35 U.S.C. § 102 Rejection - Maupin

Also, claims 1-4, 7-9, 14-15, 20, 22-23, 26-28, and 31 are rejected under 35 U.S.C. 102(b) as being anticipated by Maupin (U.S. patent No. 6,154,832, hereinafter "Maupin"). Applicant respectfully traverses these rejections and requests reconsideration in view of the following remarks.

In the present Office Action, on page 5, paragraph 13, it is suggested regarding claim 1 that Maupin taught the invention as claimed, comprising . . .

> "interrupt vector generator (e.g., see col. 5, lines 35-44) and shadow set mapping logic coupled to register sets and interrupt generator (e.g., see col. 6, lines 16-32)[task-ids assigned to interrupt sources and execution core correlates task-ID to interrupt source and using interrupt task-ID to select register set (46a-46H)."

However, Applicant submits that Maupin discloses a system readily distinguished from the presently claimed. In contrast to the presently claimed system and methods, Maupin discloses a system configured to eliminate signaling interrupt requests to a processor. To that end, Maupin discloses a processor configured to identify service requests from I/O devices by polling status registers of the respective I/O devices. For example, Maupin discloses:

> "A variety of the devices shown in FIG. 1 may rely upon processor 12 for software services which would typically be requested using one or more interrupt signals to processor 12. However, in the present embodiment, interrupt signals are not provided. Instead, each device which relies upon processor 12 for service (an interrupt source) includes a service request register 34A-34F. When a device determines that it particular service is needed, that device records the service request in the corresponding service request register 34A-34F. Meanwhile, processor 12 may continue executing a task without interruption.
>
> Processor 12 periodically polls the service request registers 34A-34F in order to determine which devices are requesting service. A task corresponding to each device which is requesting service is scheduled for execution upon polling a service request from that device. The tasks are then executed according to the schedule, and hence each device receives the requested service. By eliminating interrupt signals from processor 12, the design and verification of processor 12 may be simplified. As used herein, the term "poll" refers to a periodic read operation to determine a value stored in a storage location without modifying that storage location.
>
> Processor 12 may additionally be configured with a variety of register sets within its register file. A different register set may be permanently assigned (or "dedicated") to tasks corresponding to each of the interrupt sources within embedded controller 10. Furthermore, a register set may be dedicated to one or more interrupt sources external to embedded controller 10. Finally, a register set may be dedicated to the default task or tasks which may be executed by processor 12. Because separate register sets are assigned to each task, context save and restore operations may be eliminated. Overall performance of processor 12 may be increased due to the reduced or eliminated need for context save and restore operations." (Maupin, col. 3, line 66 – col. 4, line 36).
>
> "Service request registers 34A-34F may indicate requests for service using any suitable encoding. For example, a bit may be assigned to each possible service requested by a particular interrupt source. An interrupt service

> routine (which may comprise the task corresponding to each interrupt source) includes instructions which examine the value stored in the corresponding service request register to determine the service being requested. Generally, each service request register 34A-34F may use a different encoding, format, etc., as may be desired according to design choice. " (Maupin, col. 5, lines 34-44).

As may be seen from the above, Maupin's system is quite distinct from that presently claimed. Each of Maupin's I/O devices includes a service request register that is polled by the processor in order to determine which of the devices may require service. Accordingly, each of the devices records a service request in its respective register rather than signalling an interrupt to the processor. Maupin further discloses a plurality of register sets, with one dedicated to a default task, and each of the others dedicated to a different interrupt source. Further, the registers disclosed by Maupin are service request registers (34A-34F) that allow an interrupt service routine to determine the service being requested. However, Maupin's service request registers do not contain interrupt vectors associating with an interrupt handler. In contrast, they store a value that may be examined to determine the service being requested. For example:

> ". . . each device which relies upon processor 12 for service (an interrupt source) includes a service request register 34A-34F. When a device determines that it particular service is needed, that device records the service request in the corresponding service request register 34A-34F. " (Maupin, col. 4, lines 4-8).

Thus, Applicant finds no teaching or suggestion in Maupin of "an interrupt vector generator, for generating an exception vector associating with an interrupt handler," as is recited in claim 1. For at least this reason, each of the independent claims are distinguished from Maupin.

In addition, Maupin does not teach "shadow set mapping logic comprises a plurality of entries, each of which is programmable to associate exception vectors with at least one of said plurality of shadow register sets." Rather, each of Maupin's register sets is permanently assigned or dedicated to a task corresponding to an interrupt source or to the default task. Accordingly, Applicant submits each of the claims are patentably distinct for at least these reasons as well.

For at least the above reasons, Applicant submits each of independent claims 1, 14, 20, 23, 26, 28, and 31 are patentably distinct from Maupin. Likewise, as each of dependent claims 2-13, 15, 17-19, 22, 24, 25, 27, 30, 32, and 33-36 includes the features of the independent claims upon which it depends, each of claims 2-13, 15, 17-19, 22, 24, 25, 27, 30, 32, and 33-36 is believed patentable for at least the above reasons as well.

35 U.S.C. § 103 Rejections

In addition to the above, claims 10-13, 24, and 25 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Maupin. In addition, claims 5, 6, 17-19 are rejected under 35 U.S.C. §103 (a) as being unpatentable over Maupin as applied to claims 1-4, 14, and 15 above, and further in view of M. Morris Mano (book entitled Computer system Architecture). In view of the above remarks, Applicant submits that further traversal of these rejections is unnecessary at this time.

For at least the above reasons, Applicant submits that each of the claims is patentable over the cited art, either singly or in combination. Accordingly, Applicant believes the application to be in condition for allowance.

CONCLUSION

Applicant submits the application is in condition for allowance, and an early notice to that effect is requested.

Respectfully submitted,

/James W. Huffman/

Reg. No. 35,549
ATTORNEY FOR APPLICANT(S)

Date: 8/27/2006